

Simulation Based Engineering Laboratory  
Technical Report 2008-05  
University of Wisconsin - Madison

Scalability of Rigid Body Frictional Contacts with  
Non-primitive Collision Geometry

Justin Madsen,  
Ezekiel Kolajo

August 18<sup>th</sup>, 2008

## Abstract

Multibody Dynamic simulation involving many rigid body frictional contacts is currently difficult due to the method many commercial simulation software packages utilize. For example, the Simulation Based Engineering Lab (SBEL) regularly utilizes the popular virtual prototyping software package MSC/ADAMS, which deals with rigid body collisions using the penalty method. This method scales quadratically as the number of colliding bodies increases; therefore, it is not suitable for simulations with many rigid body frictional contacts. Engineering systems such as pharmaceutical drug packing processes, pebble bed nuclear reactors, vehicles navigating off-road terrain and conveyor belts contain many colliding bodies and currently cannot be simulated using techniques with poor scalability. When convex non-primitive shapes are used as collision objects, they can be represented by approximating the surface of the object as a triangle mesh. Non-convex shapes can be represented by combining a set of convex meshes, which allows engineering parts with very complex geometry to be modeled. In order to simulate engineering systems with many colliding objects, which may have complex collision geometry, it is necessary to use a method that solves contact forces with linear (or near linear) CPU time complexity. Recent work at SBEL has focused on a C++ SDK called Chrono::Engine, which uses a method that solves contact forces much more efficiently than the penalty method. It was observed that the CPU time required increased linearly as the number of rigid bodies (which had primitive collision geometry, i.e., spheres) increased. This technical report investigates the scalability of Chrono::Engine when non-primitive convex shapes are used as the collision geometry. Simulations run in Chrono::Engine are reproduced in ADAMS and the required CPU time to simulate a certain number of bodies is compared.

## Contents

<b>1. Introduction</b> .....	3
2. Simulation Model .....	3
<b>2.1. Geometry and Rigid Bodies</b> .....	3
<b>2.2. ADAMS/View Contacts</b> .....	5
<b>2.3. Chrono::Engine Contacts</b> .....	5
3. Simulation Parameters .....	5
4. Procedure .....	5
5. Results .....	5
<b>5.1. ADAMS/View Results</b> .....	5
<b>5.2. Chrono::Engine Results</b> .....	7
6. Conclusion .....	8
7. References .....	9

# 1. Introduction

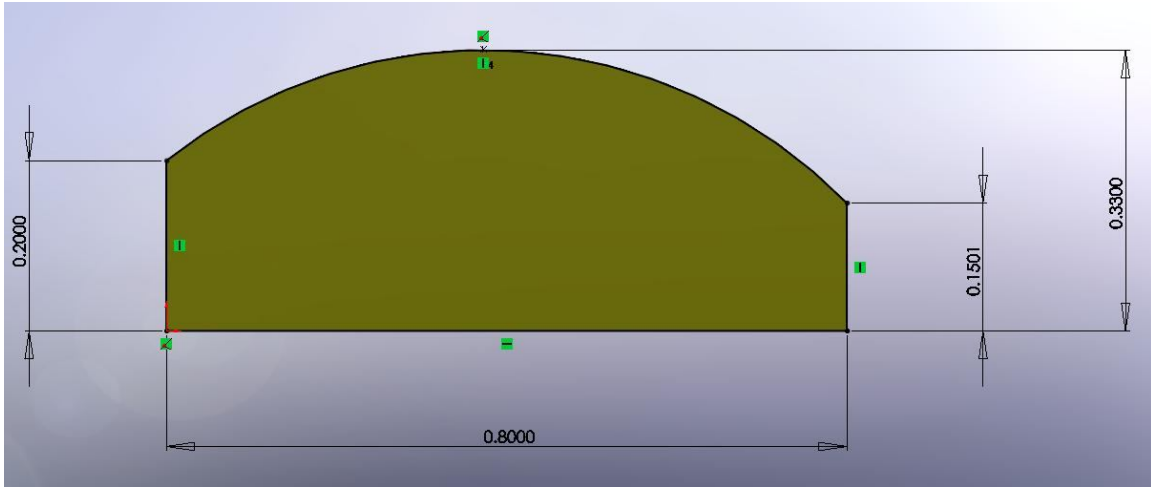
A previous technical report investigated the CPU time required to run multibody dynamic simulations that contained an increasing number of rigid body frictional contacts [1]. The simulations were run with two different virtual prototyping software programs, each which resolves collision forces using different methods. The first program, ADAMS/View, is a popular simulation program used by industry. It utilizes a penalty based approach when solving contact forces between colliding bodies, which treats the contact as a very stiff spring-damper and calculates the reaction force based on the amount of intersecting volume which violates the collision constraint. The second program, Chrono::Engine, is a Software Development Kit (SDK) that consists of a set of C++ libraries that are used to solve the equations of motion in a multibody dynamic system. This program handles contact constraints by describing them as a set of unilateral constraints in the form of a Linear Complementarity Problem (LCP) which is then solved by a fixed-point iterative scheme [2]. It was shown that when primitive shapes, in this case spheres, were used as the colliding bodies in the simulations the CPU time each software program required scaled differently as the number of bodies increased. A linear trend was observed for Chrono::Engine and a quadratic trend was observed when using ADAMS/VIEW. When the number of rigid bodies in the ADAMS/VIEW simulation increased above 32, the simulation speed nearly ground to a halt. Conversely, Chrono::Engine had no problem handling thousands of colliding rigid bodies. However, the way the bodies' geometry is handled could have a major impact on the computation time required. This is because ADAMS represents any type of geometrical shape used for 3-D collision detection as a triangular mesh; the attributes of which are determined by the third party software RAPID [3]. Chrono::Engine can represent the geometry of geometric primitives such as spheres, boxes, cylinders and tori mathematically; hence the difference in complexity could be due to the way collision geometry is represented. When collision geometry is non-primitive and convex, it is represented as a triangular mesh in both software programs. This technical report investigates how non-primitive, convex collision geometry affects the CPU time required to simulate a system with an increasing number of rigid body frictional contacts. The system model and simulation parameters are discussed, followed by the experimental procedure used. Results and conclusions are then presented and discussed.

## 2. Simulation Model

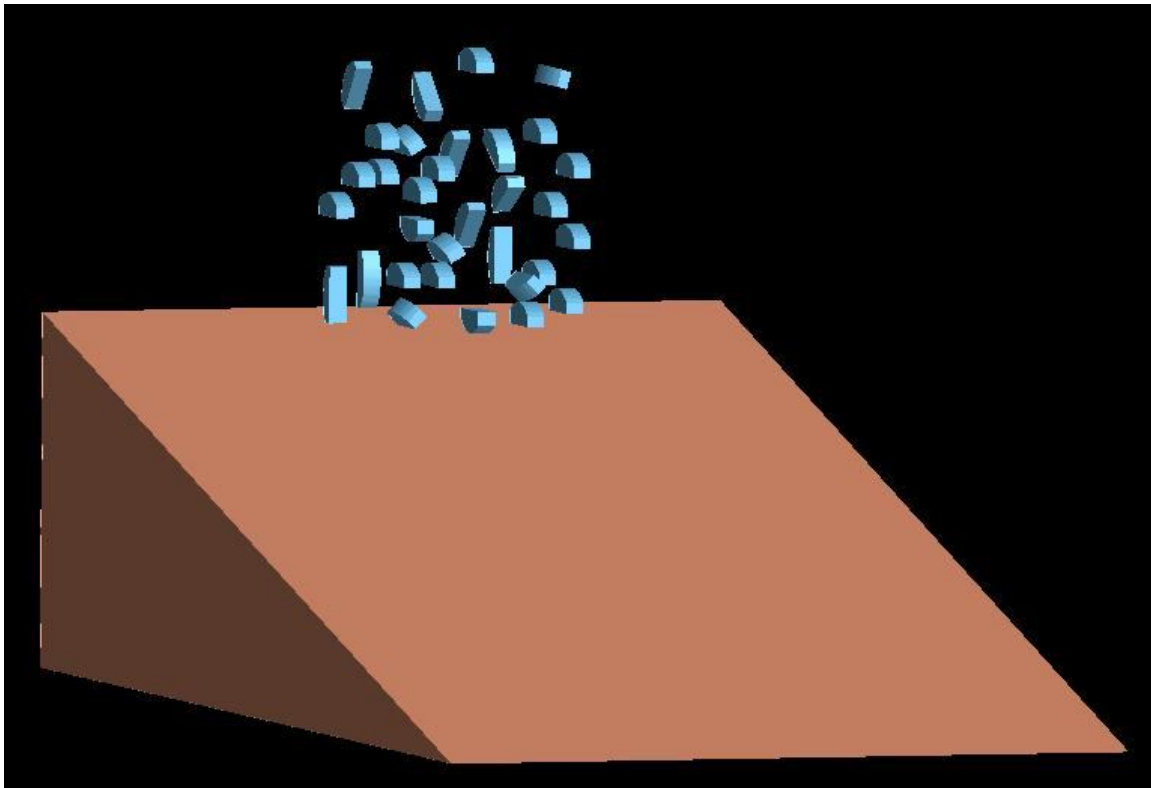
### 2.1. Geometry and Rigid Bodies

This investigation follows the procedure seen in the previous tech report [1] but uses different types of geometric shapes as collision objects. A convex, non-primitive body was created in Pro-E, a 3D CAD modeling program, and imported into ADAMS/View. Every convex shape was identical, having a thickness of 25cm, a length of 80 cm, and a maximum height of 33cm as shown in Figure 1. The shapes were placed above an inclined plane with an angle of 15.12 degrees between the top surface and the ground; the initial positions and orientations of each body were recorded. Figure 2 shows the initial configuration of the system in ADAMS/View. Each body in the simulation could collide

with any other body; however, the methods for collision detection and determination in each software program are distinctively different, hence the method describing contact forces in each program varies. Each method is described in the following sections.



**Figure 1: Collision Geometry as seen in CAD modeling program**



**Figure 2: Initial positions of objects in the ADAMS/View simulation**

## 2.2. ADAMS/View Contact Constraint

Contact forces were created in ADAMS using the impact option, with each contact having identical parameters. Contact forces were added between each shape and the incline plane with a stiffness of 1E5 N/mm, force exponent of 2.2, damping coefficient of 10.0 N-s/mm, a penetration depth of 0.1 mm and a Coulomb friction coefficient of 0.3.

## 2.3. Chrono::Engine Contact Constraint

When three dimensional rigid bodies are created in Chrono::Engine, they automatically inherit a contact force with every other rigid body in the system *unless specified otherwise*. A coefficient of restitution of 0.2 was used, as well as dynamic and static coefficients of friction of 0.3.

# 3. Simulation Parameters

Each simulation run in ADAMS was carried out twice using the default integrator type (GSTIFF). First, a simulation with a time step size of 0.01 seconds was used followed by a simulation with a time step size of 0.001 seconds. Chrono::Engine used its default integrator type (INT\_TASORA) with a timestep of 0.001 seconds. All simulations were run for 3 seconds of simulation time.

# 4. Procedure

Each simulation was run twice in ADAMS (timestep varied between 0.01 and 0.001 seconds) and five times in Chrono::Engine (timestep of 0.001 seconds) where the CPU time required was averaged from these five simulations. Every simulation was allowed to run until 3 seconds of simulation time was reached and the total CPU time required was then recorded. These steps were repeated for six different simulations with varying numbers of convex bodies present in the system—1, 2, 4, 8, 16, and 32 bodies respectively.

# 5. Results

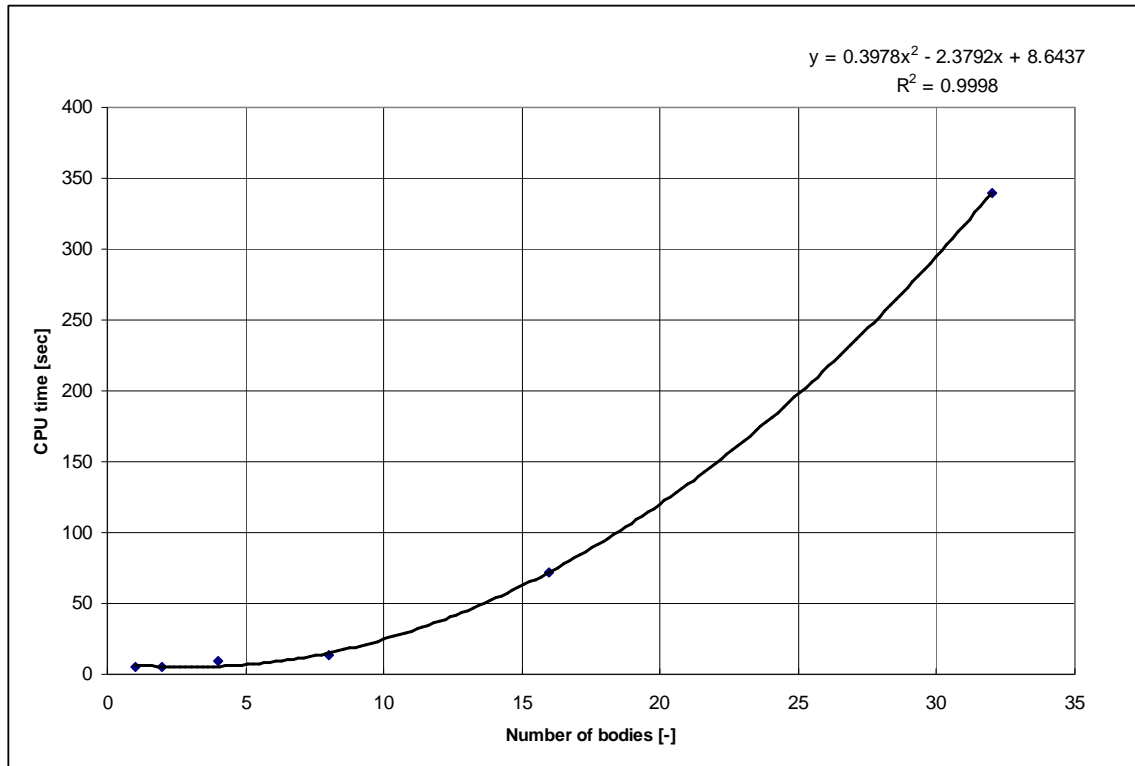
## 5.1. ADAMS/View Results

The following results show the CPU time required for the ADAMS/View simulations. In the first series of simulations, run with a timestep of 0.01 seconds, it can be seen in Table 1 that a fairly rapid increase in CPU time occurred as the number of bodies increased.

**Table 1: Number of convex shapes & CPU time in ADAMS/View (timestep = 0.01sec.)**

Number of Bodies [-]	CPU time [seconds]
1	5
2	5
4	9
8	14
16	72
32	340

This nonlinear increase in CPU time can be better represented by plotting CPU time v. number of colliding bodies, seen in Figure 3. As the figure illustrates, CPU time increases quadratically with the number of colliding bodies.



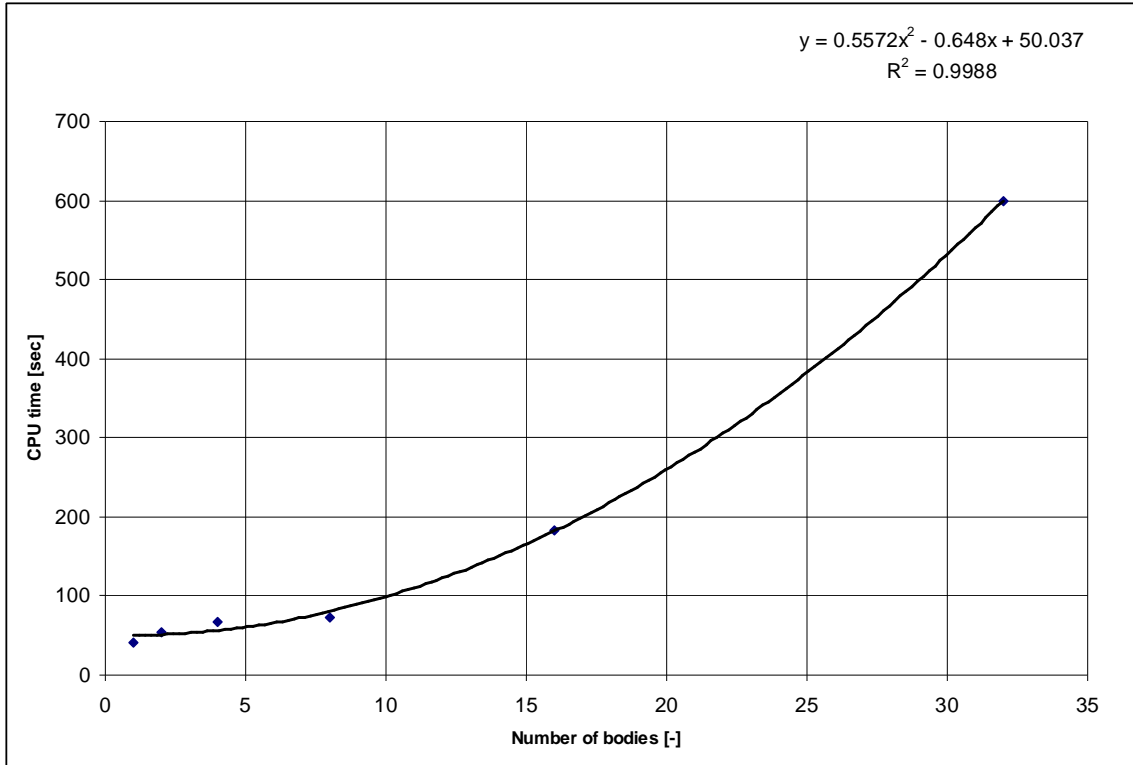
**Figure 3: Nonlinear (quadratic) increase of CPU time in ADAMS/View  $\Delta t = 0.01$  sec**

Similar results were obtained for the simulations run with a step size of 0.001 seconds. As seen in Table 2, the increase in CPU time followed a similar trend; however, the simulation times were much larger for each corresponding simulation.

**Table 2: Number of convex bodies & CPU time in ADAMS/View (timestep = 0.001sec.)**

Number of Bodies [-]	CPU time [seconds]
1	41
2	55
4	68
8	73
16	183
32	600

Once again, quadratic complexity can be observed in Figure 4 when the data in Table 2 is plotted.



**Figure 4: Nonlinear increase of CPU time in ADAMS/View,  $\Delta t = 0.001$  sec**

The large differences seen in the CPU time for corresponding trials in both simulations is due to the different timestep sizes used; however, the complexity is independent of the timestep selected.

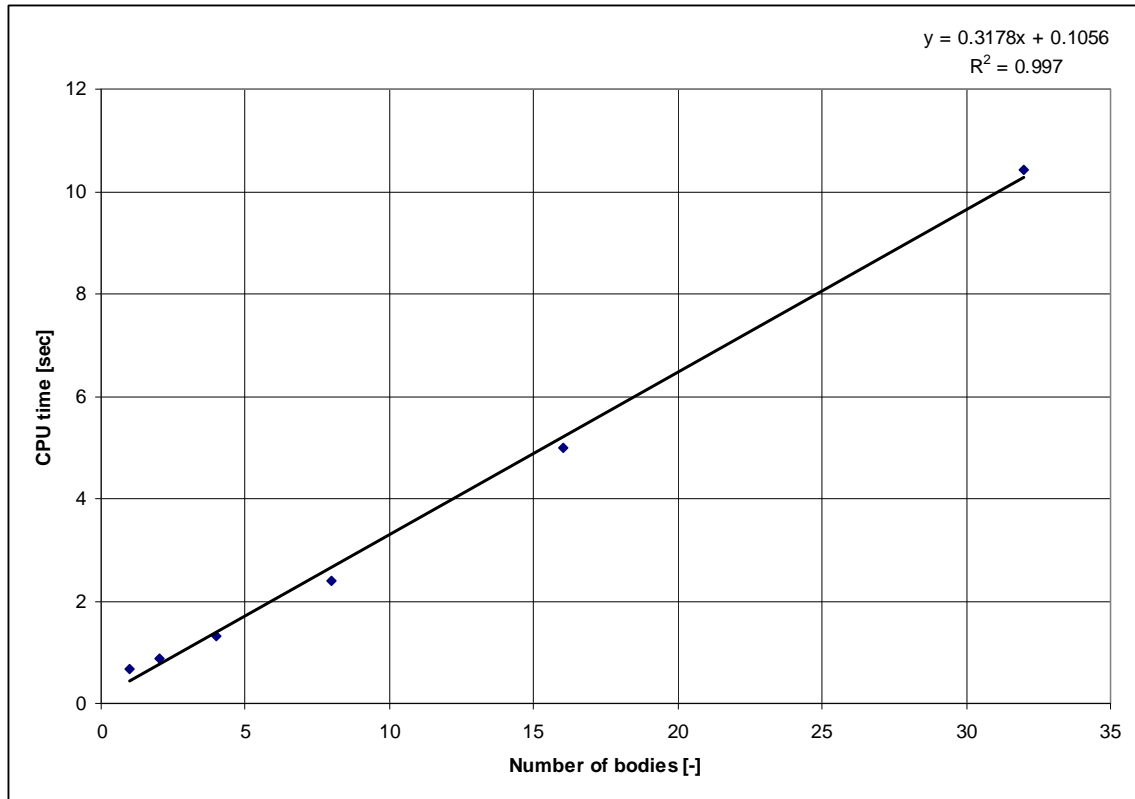
## 5.2. Chrono::Engine Results

Results of the Chrono::Engine simulations can be seen in Table 3. Because five simulations were run for each system with a certain number of colliding bodies, the standard deviation can also be included.

**Table 3: Number of convex bodies, CPU time & Standard Deviation in Chrono::Engine**

Number of Bodies [-]	CPU time [seconds]	Standard deviation [sec]
1	0.6585	0.00262
2	0.8793	0.00402
4	1.3066	0.00631
8	2.3941	0.00518
16	4.9804	0.00756
32	10.4370	0.00416

Plotting CPU time v. the number of colliding bodies shows a nearly linear increase in CPU time as the number of colliding bodies increases (Figure 5).



**Figure 5: Linear increase of CPU time in Chrono::Engine**

## 6. Conclusion

This technical report investigated how the number of colliding rigid bodies with non-primitive geometry affects the CPU time required for a simulation when two different methods were used to describe and solve the contact forces. A penalty based approach is used in the software program ADAMS/View and a quadratic increase in CPU time was observed as the number of colliding bodies increased (Figures 3,4). When the contacts are described as unilateral constraints, complementarity inequalities can be introduced to form a linear complementarity problem. A fixed-point iterative method, found in the C++ SDK Chrono::Engine, is used to solve the LCP. This resulted in a nearly linear increase in CPU time as the number of colliding bodies increased (Figure 5). It is important to note that convex shapes can be used to model many more relevant engineering parts than primitive geometry is able to, and many convex shapes can be combined to form even more complex geometry (i.e., concave geometry). This is ideal when dealing with contact heavy simulations that have complex collision geometry, such as in [4]. Reducing the computational overhead for contact heavy simulations will pave the way for accurate and time-efficient simulations for such processes as powder packing, conveyor belts, engine belts, pebble bed nuclear reactors and any type of granular flow (i.e., sand and gravel simulation). Future work will investigate the feasibility of using the methods for handling contacts found in Chrono::Engine in contact heavy simulations involving complex engineering machinery such as NASA rovers [5] and hydraulic excavators [4].

## 7. References

- [1] J. Madsen, N. Pechdimaljian, D. Negrut, *Penalty Versus Complementary-Based Frictional Contact of Rigid Spheres: a CPU Time Comparison*. SBEL technical report TR-2007-06, <http://sbel.wisc.edu/>, University of Wisconsin – Madison.
- [2] Tasora, A., *An iterative fixed-point method for solving large complementarity problems in multibody systems*. GIMC 2006, XVI Congress of the Italian Group of Computational Mechanics, 26-28 June 2006, Bologna – Italy, 2006.
- [3] Gottschalk, S., Lin, M.C., Manocha, D., *OBB-Tree: A hierarchical Structure for Rapid Interference Geometry*. Department of Computer Science, Technical report TR96-013, University of North Carolina Chapel Hill.
- [4] Madsen, J., *High Fidelity Modeling and Simulation of Tracked Elements for Off-Road Applications Using ADAMS/VIEW*. SBEL technical report TR-2007-02, <http://sbel.wisc.edu/>, University of Wisconsin – Madison..
- [5] Mars Exploration Rover Mission: Home. <http://marsrovers.nasa.gov/home/>